Virtual Wind-up Toys for Animation

Michiel van de Panne
*Ryan Kim**
Eugene Fiume

Department of Computer Science and *Electrical Engineering
University of Toronto, Toronto, Canada, M5S 1A4†
email: {van | rkim | elf}@dgp.utoronto.ca

Abstract

We propose a new method of automatically finding periodic modes of locomotion for arbitrary articulated figures. *Cyclic pose control graphs* are used as our control representation. These specifically constrain the controller synthesis process to only those controllers producing periodic driving functions. It is shown that stochastic generate-and-test techniques work well with this representation. Several choices that arise in this synthesis technique are explored. The impact of the design of the physical models upon the motions produced is examined. Lastly, the motions produced are analysed by looking at their bifurcation diagrams.

Keywords: animation, control, simulation, locomotion, modelling, limit cycles

Résumé

Nous présentons une nouvelle méthode permettant de trouver automatiquement des mouvement periodiques pour des figures articulées. Nous utilisons des réseaux cycliques de poses pour représenter et contrôler les mouvements. Ces réseaux contraignent la synthèse des contrôleurs à ne produire que des contrôleurs générant des mouvement périodiques. En utilisant cette répésantation, des contrôleurs sont générés de façon aléatoire, puis mis à l'épreuve. Nous examinons quelques variations qui existent dans la technique de synthèse. Nous vérifions aussi comment la construction du modèle physique luimême affecte les mouvements produits. Dernièrement, nous analysons le phénomène de bifurcation dans ces mouvements.

1 Introduction

One of the promises of using physical simulations in creating animations is that they can be used to simplify the design of motions. A physical simulation of an object's motion ensures that all the constraints imposed by the laws of physics are enforced. For *active* systems, however, we must also solve the additional problem of how to control a creature's muscles in order to obtain a desired motion. It is this *control problem* which is the specific focus of our research.

Many techniques have been proposed for controlling the motion of a variety of creatures. Much research has focussed on the specific problem of controlling modes of locomotion, and our results are no exception in this regard. Perhaps the most striking feature of almost any mode of locomotion is that it is periodic. We shall show that by constraining our search to those controllers that produce periodic driving functions, we can efficiently find and optimize many modes of locomotion.

Good results have been obtained by constructing controllers that explicitly produce periodic outputs. Miller used periodic sinusoidal contractions and expansions to obtain modes of locomotion for physically-based models of worms and snakes[10]. McKenna and Zeltzer used a set of coordinated periodic motions to obtain robust walking motions for a model of a cockroach[9]. Van de Panne, Fiume, and Vranesic use variations of periodic control to balance and steer turning figures[18]. The technique in this paper presents a way of *automating* the synthesis of periodic controllers.

The work presented in this paper builds on some of the ideas presented by Hodgins et al. in [5]. In this work, state-machines are used to implicitly specify periodic motions, including riding a see-saw, pumping a swing, and juggling three balls in various patterns.

Other types of hand-programmed controllers have also had success in controlling periodic motions. Raibert and Hodgins present an elegant method for controlling periodic hopping gaits for a variety of creatures[13]. Stewart and Cremer present a walking controller based upon the addition and removal of constraints[14]. Bruderlin and

†The financial assistance of the Natural Sciences and Engineering Research Council of Canada, and of the Information Technology Research Centre of Ontario, is gratefully acknowledged.

Calvert construct a walking controller by applying knowledge of the state-phase timing[2]. We also build on the early ideas of Isaac and Cohen[6].

The controllers outlined above have a large hand-designed component, which is an impediment to the application of these control techniques to arbitrary systems. Optimization techniques provide some promise of automating the design of controllers. Witkin and Kass[19] (and later expanded upon by Cohen[3]) propose a powerful technique for optimizing motions as trajectories over time. One restriction of dealing with motions as trajectories is that it is difficult to properly incorporate interactions with the environment. Discontinuities in the motion, such as those caused by impact with the ground, pose difficulties for many optimization techniques. Other optimal control techniques have also been applied by Girard[4], Pandy et al.[12], and van de Panne et al.[16]. The size of the search space of most optimization techniques tends to grow quickly as a function of the complexity of the object.

An alternative approach is to use a generate-and-test optimization process. This kind of approach is founded upon the idea that it is relatively easy to determine if a *given* controller produces a desired motion. While this is the inverse of the problem that we are trying solve, it can be used as the basis of a synthesis process. This is done by using repeated trials of a controller while changing the parameter values defining the controller. An evaluation metric, such as the distance travelled, can then be used to determine whether a given parameter change contributes towards a desired behaviour or not. Two variations on this type of approach were investigated independently by Ngo and Marks[11], and van de Panne and Fiume[17]. The algorithm is perhaps best called *generate-and-test* when controller parameters are chosen completely at random, and *modify-and-test* when modifications are being made to an initial set of parameter values.

This paper will make use of the generate-and-test and modify-and-test ideas, but with a simpler control representation than those discussed in [11] and [17]. The pose control graphs we shall discuss do not make use of any *sensory* information, unlike the representations in [11] and [17]. We shall perform an investigation into the performance of controllers that use only fixed, timed transitions between states or *poses*. Pose control graphs thus provide *open-loop* control. It is only after investigating simple control schemes without sensors that we can assess what and how sensors can contribute to constructing controllers. Future extensions could involve using sensory information to perform state transitions when necessary.

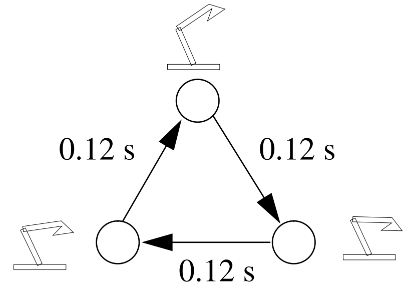


FIGURE 1. A pose control graph for Luxo, an articulated, hopping lamp. The pose for each state defines a desired internal configuration. Transitions between states happen after the time intervals indicated on the arcs.

2 Pose Control Graphs

A pose control graph is a state-machine that has a particular *pose* associated with each state. An example is shown in Figure 1. The pose represents the desired internal configuration of a creature when the controller is in a particular state on the control graph. A pose thus specifies the desired *shape* of the creature, but not its position or orientation in the world. The position and orientation of the body in the world are determined by the interaction of the creature with its environment. A pose creates torques to drive the shape of the creature towards that specified by the pose. These torques are calculated by using proportional-derivative (PD) controllers placed at each joint. The torques produced are given by $\tau = k_s(\theta_d - \theta) - k_d\dot{\theta}$, where θ_d is the desired relative angle between two links, and the constants k_s and k_d determine the contributions of the proportional and derivative terms. Torques are usually largest immediately after a transition to a new pose has taken place.

A change of state is performed in a pose control graph after the controller has rested in a given state for the amount of time specified by the exiting arc. In this paper we shall deal only with the simplest of pose graphs, those which have a cyclical sequence of timed transitions.

Pose control graphs are not an entirely new concept in themselves. Hodgins et al. make extensive use of this kind of structure in [5]. Our investigation into the automatic synthesis of cyclic, timed pose control graphs is new, however. Ngo and Marks use poses to control the actual shape of the creature rather than its desired shape, and thus do not use a true physical basis for governing the internal shape of the creature[11]. Van de Panne and Fiume use the concept of a desired pose, but have a distributed representation of the state of the controller based upon a network of connections[17].

A pose control graph with timed transitions produces open-loop control. In effect, we are dealing with wind-up toys which have no awareness of their environment. Despite this seemingly severe limitation, the mechanics of a creature interacting with the ground can lead to dynamically stable motions for many walking and hopping creatures. It can thus be demonstrated that many interesting modes of locomotion are achievable with the equivalent of dumb "wind-up toys".

We have experimented with two particular variations of pose control graphs. The first holds the desired pose fixed for the duration of a given state in the pose control graph. The second investigates the alternative of allowing the desired pose to vary linearly between that of the current state and that of the next state. This variation takes place during the time interval specified by the transition arc between the current and next states.

Our synthesis experiments indicate that there is little difference between these two variations in their capability to produce interesting and useful controllers. Figure 2 shows the operation of the first variant, where step changes in the desired poses are allowed. Figure 3 shows the operation of the same joint using linear interpolation between desired poses. In general, allowing for step-changes in the desired pose leads to more energetic motions than the linear variation of desired poses, if both types of poses are restricted to identical ranges. We did not perceive the motions produced by one method as being more natural-looking than the other, although this is perhaps because the torques produced never became excessively large in our experiments.

3 Synthesis of Pose Control Graphs

An ideal automated synthesis system would be able to design an efficient locomotion controller given only the mechanical structure of the creature (including actuators) and no other *a priori* information. For cyclic pose control graphs, it is necessary to specify a small but useful amount of additional information that can greatly reduce the synthesis or search time. The information required is an estimate of the period duration and an estimate of the number of poses required for the motion. These two numbers are usually easy to estimate. If necessary, they could be derived from a keyframed version of the motion or video data. An estimate of the period duration might be determined based upon the size of a creature.

Perhaps the single most important property of the pose control graphs is that they are cyclic and thus always produce a periodic control function of the desired frequency. This greatly reduces the search space for possible control strategies. In the work of Ngo and Marks[11] and van de Panne and Fiume[17], periodic motions are often produced as being the best solutions, but there are no explicit constraints which restrict the search to peri-

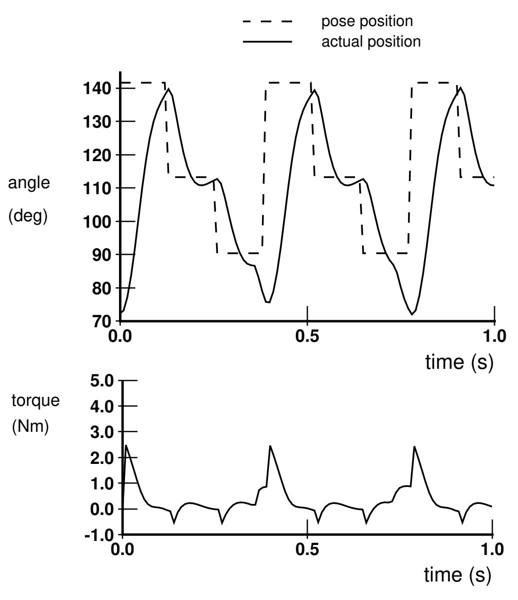


FIGURE 2. (a) Desired and actual position of joint A2 of Luxo during a hopping gait. In this case, the desired pose remains fixed while in a particular state of the control graph. The pose control graph used is shown in Figure 1. The effect of the impact of the base with the ground can also be seen. (b) The internal torque being produced at this joint

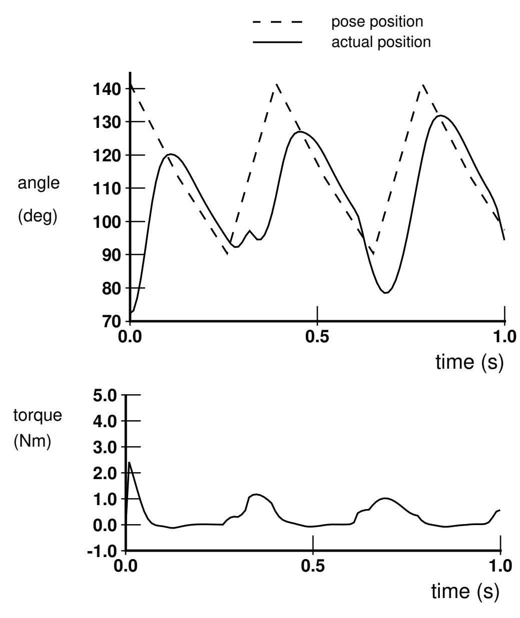
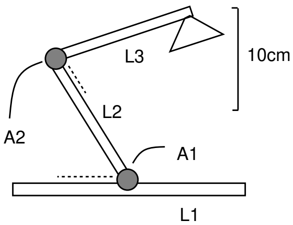


FIGURE 3. (a) Desired and actual position of the same pose-control graph as in Figure 2, but with linear variation of the desired pose. (b) The internal torque being produced at this joint. In general, the motion is not as energetic, given the same pose-control.

mechanical configuration:



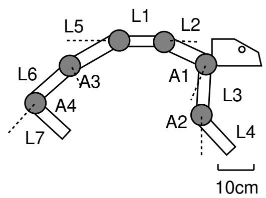
link	mass (kg)
L1	0.05
L2	0.10
L3	0.30

actuators:

act.	min	max	ks	kd
A1	−70	−50	0.05	0.001
A2	60	150	0.04	0.001

FIGURE 4. Luxo, the hopping lamp.

mechanical configuration:



link	mass (kg)
L1	0.30
L2	0.35
L3	0.15
L4	0.10
L5	0.35
L6	0.15
L7	0.10

actuators:

act.	min	max	ks	kd
A1	−45	45	0.4	0.01
A2	−90	0	0.4	0.01
A3	40	70	0.4	0.01
A4	−120	−30	0.4	0.01

FIGURE 5. The cheetah creature. There are two joints in the back which have passive spring and dampers. Making the back flexible and allowing it to act as a spring capable of storing energy is important in obtaining natural running motions.

odic solutions. The search space in these methods consists of pairings of sensory information to various possible actions. The best such pairings are often the ones which lead to periodic motions. In these cases, the periodic motion arises *implicitly* out of a stimulus triggering a response, eventually leading to the triggering of a new response, and so on. A cyclic pose control graph representation avoids spending time searching through aperiodic solutions and investigates to what extent one can synthesize useful motions without using sensory information.

Before any controller synthesis can be done, a mechanical model of the creature is required. The examples we shall be working with in this paper are Luxo, cheetah, and the bounder, shown in Figures 4, 5, and 8, respectively. The design includes the specification of the range of operation for the joints and the PD constants associated with the joints. The joint ranges are used for limiting the range of the control graph poses. The PD constants in effect specify the strength of actuators. Given the desired period of the motion, T, and the number of poses, n, the cyclic pose graph will consist of n poses, with timed-transistions of duration T/n. The synthesis technique must then find the n poses that will perform well with respect to a given optimization metric. The question of how to choose an optimization metric that leads to a desired motion is an interesting but difficult one. We shall not dwell on it here. For our examples, we choose to use distance travalled in 4 seconds as an optimization metric.

As with our previous work with sensor-actuator networks[17], we shall make use of generate-and-test and modify-and-test algorithms for synthesis of pose control graphs. This requires the use of an efficient simulator; we use an optimized 2D mechanical simulator[17]. Extending our control ideas to 3D requires using an appropriate 3D simulator. The difficulty of a control problem is not necessarily related to its dimension, however.

The synthesis procedure we use is largely the same as that described in [17], only we now apply it to a different control representation. We use a two phase procedure. The first phase consists of random generation and evaluation of controllers. A pose control graph is generated by choosing the n poses at random. The pose control graph is then evaluated with respect to the optimization metric. For our examples, this involves placing the figure at rest in a fixed initial position and then simulating its motion when driven by the pose control graph for 4 seconds. The purpose of the first phase is to perform a global search for possible modes of locomotion. If multiple possible modes of locomotion appear as a result, the animator can choose which ones are passed on to the second phase described below. Alternatively, the best-performing modes can be automatically chosen.

The purpose of the second phase is to perform a local optimization of a gait. The basic strategy here is one of making a small change to one of the controller parameters (i.e., to one angle of a pose) and seeing if this improves the motion of the creature or makes it worse. If the change is for the better, the change is kept; otherwise it is rejected. There are many different variations which can be used in implementing this fine-tuning strategy, and we have investigated several of them.

One choice to be made is the size of the parameter change to be made for each trial. Figure 6 shows the results of three different strategies applied in fine-tuning the performance of the bounder creature (see Figure 8). For each of the three strategies, it shows the total improvement in the evaluation metric over 6 repeated runs of 500 trials. One of the simplest strategies is to modify only one parameter at a time, but to allow this parameter to change by a random amount within the allowable (but fixed) range of the joint involved. A second strategy is to use a fixed value of delta which decreases in size over subsequent trials as the optimization proceeds. In this case, only the choice of parameter and the sign of the parameter change are randomly cho-

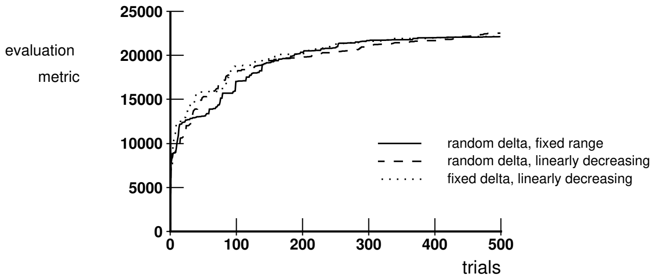


FIGURE 6. Three different strategies for choosing the size of attempted parameter modifications for the bounder creature

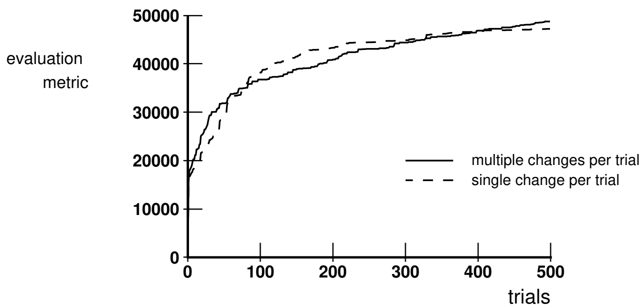


FIGURE 7. A comparison of multiple vs. single parameter changes per trial for controller synthesis for the bounder creature.

sen. A third strategy is to allow a random change chosen to lie within the decreasing range as described by the second strategy.

Surprisingly, the choice of strategy does not seem to have a great impact upon the final performance attained by the creature. When using a large fixed range, larger parameter changes are attempted and as a result they are more likely to be rejected. When such a change is successful, however, the payoff is large because it is equivalent to making multiple smaller changes. Choosing a random value of delta within a linearly decreasing range seems to be the best strategy by a narrow margin.

One can also consider the possibility of changing multiple parameters per trial in the hope of speeding up the optimization process. We performed 12 simulations of 500 trials with the bounder creature, using random-delta, linearly decreasing parameter changes, in order to evaluate the effect of performing multiple parameter changes for each successive trial. The results are shown in Figure 7. Performing multiple parameter changes proved to be marginally better in this case.

The results of the various synthesis experiments seem to indicate that the modify-and-test algorithm is not very sensitive to the method of choosing the parameter changes. Results obtained using modify-and-test algorithm seem in general to be robust and repeatable. Another variation that we have experimented with is simulated annealing (i.e., occasionally accepting changes for the worse) in order to find more global optima. The algorithm used is similar to that used in [17], only it is applied to the pose control graph representation. Our experience here to date suggests that the results here do exhibit sensitivity to the choice of annealing schedule, but can yield superior solutions. Figure 9 is an example gait produced using simulated annealing

4 Modeling Considerations

In using physically-based animation, there are many choices that influence the resulting motions. Some relate to the control representation and the optimization techniques used, but the physical model itself is also a strong determinant of the motion. The 'optimal' way our creatures can move is very much a function the way their bodies were designed. Both the 'skeleton' and the 'muscles' are important in this respect. The skeleton determines the topology of connection between the links and the distribution of mass. The 'muscles' are effectively determined by specifications of the angle ranges allowed in the pose specification, and the spring and damper constants associated with a joint.

The cheetah creature provides a useful example of how the design of the physical model can greatly affect the resulting motions. The goal here is to obtain natural-looking, planar running motions. The bounder creature, shown in Figure 8, was used in our first attempts at obtaining running motions. The motions produced for this creature were not as natural as we would have hoped. The cheetah creature shown in Figure 5 is a redesigned version of the bounder creature, having a flexible back. The back has no active muscles, but has two joints, each with a passive spring and damper to provide restoring forces when the back is deformed. The masses assigned to the various links of the cheetah are not from biomechanical data, but are chosen to reflect a reasonable distribution of the total mass. Figure 9 shows the running motion obtained for the cheetah. The flexible back plays an important role in producing this more graceful running gait

In order to assess how the flexible back benefits the running motion, it is useful to examine how the gait changes as the back becomes stiff. Figure 10 shows the same gait as in Figure 9, only using a stiff back. The stiff back is created by increasing the damping constant at the passive joints in the back. The gait is produced using the same pose-control graph.

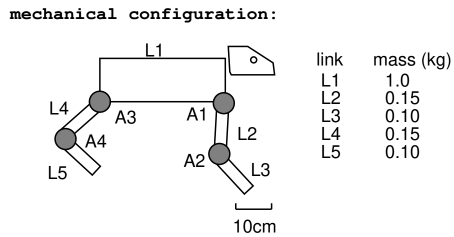


FIGURE 8. The bounder creature. The back is modelled as a single rigid link. The actuators have similar ranges and strengths to those specified for the cheetah.

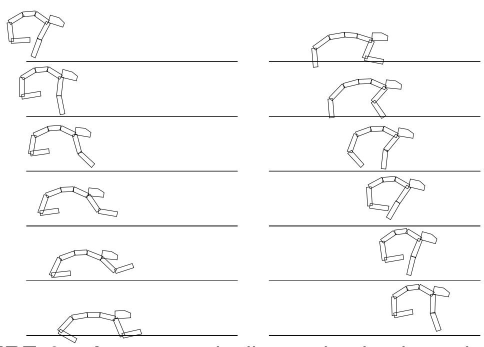


FIGURE 9. An automatically synthesized running gait using pose control graphs, for the cheetah creature with a flexible back. The animation should be read from top to bottom, then left to right.

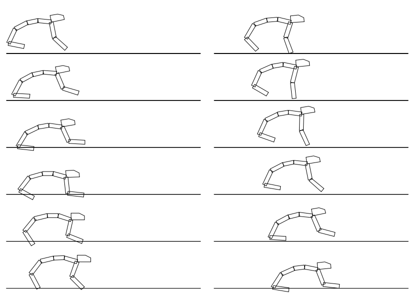


FIGURE 10. The cheetah creature with a stiff back. This gait is produced by the same pose control graph as Figure 9, only the back of the cheetah has been stiffened.

Figure 11 shows an analysis of the resulting motions. As the back is stiffened, it performs less mechanical work. At the nominal value of $k_d = 0.004$, the back is responsible for approximately 28% of the mechanical work performed in the system. This decreases as the back stiffens, reaching zero as the back approximates a rigid link. In evaluating the mechanical work, we use a modified definition that does not distinguish between positive and negative mechanical work. From Figure 11, it can be seen that the speed suffers with a stiffer back.

It is also interesting to look at the effect that the back has on an 'energy' or effort metric. Figure 11 shows that the

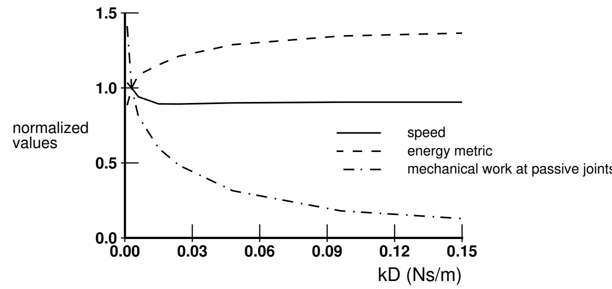


FIGURE 11. Effects of stiffening the back upon the running performance of the cheetah creature. All the values are normalized with respect to the performance obtained for the nominal value of kd=0.004, for which the pose-control graph was designed.

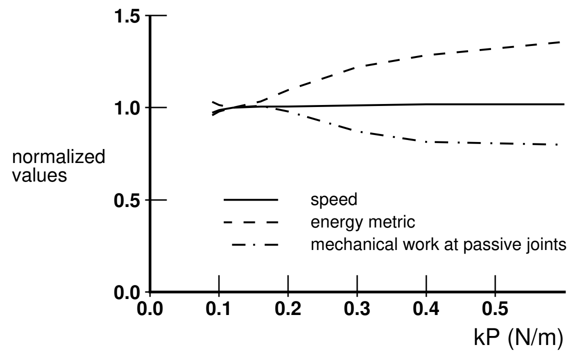


FIGURE 12. Effects of increasing kp upon the running performance of the cheetah creature. All the values are normalized with respect to the performance obtained for the nominal value of kp=0.12, for which the pose-control graph was designed.

energy metric increases by up to 33% as the back becomes stiffer. We chose $\int \tau^2$ as our metric, where τ is the joint torque, although similar results were obtained using other 'energy' or effort metrics. A damping factor even smaller than that used in the nominal design produces the best results, giving a faster and more energy-efficient gait. In this case, almost all of the energy stored in the spring elements of the passive back is reused in the motion.

In a similar fashion, we can also analyse the effect of changing the spring constant associated with the passive joints in the back, as shown in Figure 12. For $k_p < 0.1$, no results are shown because for this parameter range the cheetah creature trips and falls over. It is interesting to observe that the nominal spring parameter of 0.12 is optimal for the pose-control graph, with respect to speed, 'energy' or effort, and the amount of mechanical work performed by the back. The synthesis process has thus produced a pose-control graph specific to the physical parameters of the original model.

Including flexible backs and other energy-storage mechanisms in the physical models can be an important factor in obtaining natural motions[1]. In general, the motions produced using optimization methods are an equal product of the physical model, the optimization metric used, and the search strategy.

5 Motion Analysis

Do cyclic pose-control graphs always produce periodic motions? The answer to this question is ‘no’, despite the fact that cyclic pose-control graphs drive the creatures in a periodic fashion. The control provided is open-loop and thus relies on the interactions between the creature and the environment in order to reach a steady-state periodic motion or limit cycle. The tendency of some mechanical systems to reach periodic limit cycles has been used in the past as a useful design property[8]. Many non-linear dynamical systems are liable to bifurcations and chaotic motions, however, and the creatures we are dealing with here prove to be no exception. Some interesting previous work on bifurcations and chaotic motion analysis exists for simplified models of hopping robots[7][15].

We shall use the transition time between two states of a pose-control graph as a *bifurcation parameter* to illustrate the various types of motion that can result from a single pose-control graphs. Any single parameter of the pose-control graph can be chosen in general for such experiments. We shall examine how the behaviour of the resulting motion for a creature changes as we change the bifurcation parameter. The various phase-space diagrams to be introduced shortly all result from an identical pose-control graph for Luxo, with the exception of the bifurcation parameter, T, which we allow to vary.

Figure 13 shows *phase diagrams* for 3 different values of T. The state-space for Luxo is 10-dimensional, but one can use a projection of the state trajectory onto the plane defined by any two of the 10 dimensions in order to visualise the behaviour of the system. The figure plots the height of the centre of the base versus the angle of the base with respect to the ground. Note that because of our spring-and-damper ground model, the base sinks slightly into the ground on each hop.

For large values of the bifurcation parameter ($T > 0.14$) the motion obtained has the same period as that of the pose-control graph. Figure 13(a) shows such a periodic motion.The cycle shown is an attractor towards which nearby trajectories will converge.

As the bifurcation parameter is decreased below 0.14, a motion having twice the period of the pose-control graph emerges. The gait itself has the appearance of a limping gait, with every second hop being identical. The phase diagram is as shown in Figure 13(b), where a second loop has broken off from the first as the bifurcation occurs.

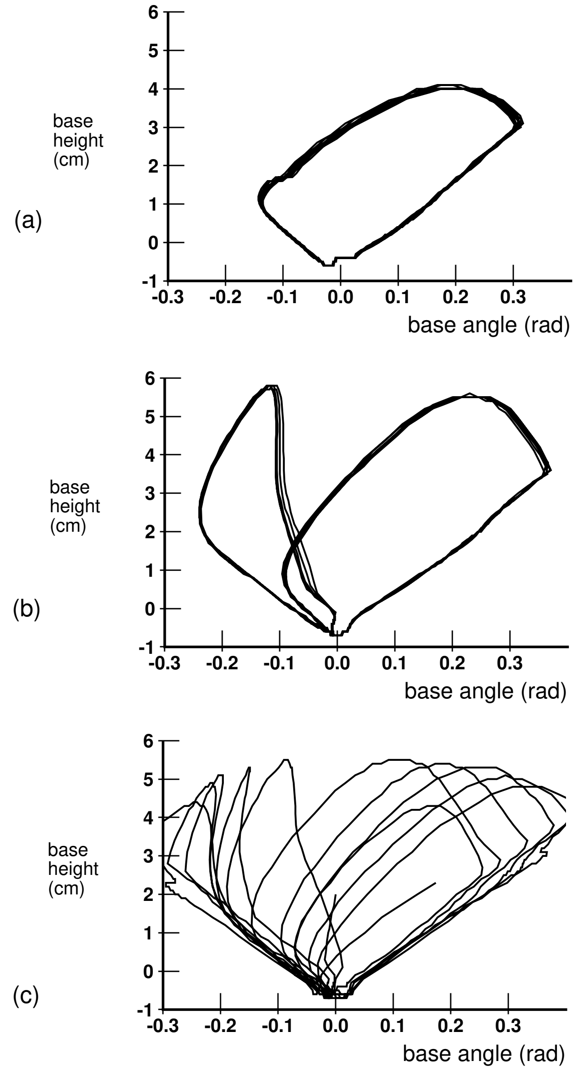


FIGURE 13. Phase diagrams for various values of the bifurcation parameter. (a) T=0.155, (b) T=0.115, (c) T=0.100.

Upon further decreasing the bifurcation parameter, a chaotic, non-repeating motion is obtained. For our particular example this region is approximately defined by $0.070 < T < 0.102$. The phase diagram shows no detectable periodicity, even over long time intervals. A typical phase diagram is shown in Figure 13(c).

Surprisingly, a regular period-two behaviour reappears for $0.046 < T < 0.070$, and a period-one behaviour reappears for $T < 0.045$. The complexity of motions that can be obtained using a cyclic pose-control graph should thus not be underestimated. The possibility of period-doubling and chaotic behaviours impacts on any synthesis strategy in several ways. First, the length of the trials used in a generate-and-test strategy should be sufficient to detect whether a convergence to a periodic motion is obtained. Chaotic motions do not produce the repeatable behaviour which is desirable for producing animations. Second, limping behaviours can appear naturally in many cases as a result of making small parameter changes. Lastly, it might be possible to remove (or perhaps add) chaotic modes to a system by allowing for transitions based upon sensory data.

6 Conclusions

Pose-control graphs are a simple control representation for which we have introduced a synthesis technique capable of automatically producing optimized periodic gaits. It unites the use of a discrete state-machine control representation with generate-and-test and modify-and-test optimization techniques. The 'wind-up toy' idea they embody is simple, yet powerful enough to produce natural-looking running gaits. They are different from previous work which relies on a chain of sensing-action (stimulus-response) events[11][17]. Pose-control graphs are in some sense simpler in that periodic motions can be explicitly specified, and their structure can later be extended to include the use of sensory information *when it is needed*. From the point of view of animators, pose-control graphs can be seen as a natural extention of keyframing to a physically-based setting.

Our experiments with obtaining running behaviours for the cheetah indicate that constructing appropriate physical models is important in obtaining desireable motions. Passive, elastic elements such as flexible, springy backs seem to be important for natural and efficient motion. Eventually, one can perhaps hope to compare synthesized motions with the real motions of animals, given that a suitable model has been constructed.

Period-doubling bifurcations and chaotic behaviour are phenomena which can easily occur during the synthesis process. Analysis of the phase diagrams is useful in revealing an underlying structure in many motions, and can perhaps later be used to justify the addition of sensory feedback into the pose-control graph.

References

[1] R. M. Alexander. *Elastic Mechanisms in Animal Movement*. Cambridge University Press, 1988.

[2] A. Bruderlin and T. W. Calvert. Goal-Directed Animation of Human Walking. *Proceedings of SIGGRAPH '89*. In *ACM Computer Graphics,* 23(4), 233-242, 1989.

[3] M. F. Cohen. Interactive Spacetime Control for Animation. *Proceedings of SIGGRAPH '92*. In *ACM Computer Graphics*, 26, 2 (July 1992), 293-302.

[4] M. Girard. Constrained Optimization of Articulated Animal Movement in Computer Animation. In *Making Them Move*, Morgan Kaufmann, 1991, 209-232.

[5] J. K. Hodgins, P. K. Sweeney, and D. G. Lawrence. Generating Natural-looking Motion for Computer Animation. *Proceedings of Graphics Interface '92*, 265-272, May 1992.

[6] P. M. Isaac and M. F. Cohen. Controlling dynamic simulation with kinematic constraints, behavior functions, and inverse dynamics. Proceedings of SIGGRAPH '87. In ACM Computer Graphics, 21, 4 (July 1987), 215-224.

[7] D. E. Koditschek and M. Buhler. Analysis of a Simplified Hopping Robot. *The International Journal of Robotics Research*, 10, 6, 1991, 587-605.

[8] T. McGeer. Passive Dynamic Walking. *The International Journal of Robotics Research*, 9, 2, 1990, 62-82.

[9] M. McKenna and D. Zeltzer. Dynamic Simulation of Autonomous Legged Locomotion. *Proceedings of SIGGRAPH '90.* In *ACM Computer Graphics*, 24, 4 (August 1990), 29-38.

[10] G. S. P. Miller. The Motion Dynamics of Snakes and Worms. *Proceedings of SIGGRAPH '88.* In *ACM Computer Graphics*, 22, 4 (August 1988), 169-178.

[11] J. T. Ngo and J. Marks. Spacetime Constraints Revisitied. Proceedings of SIGGRAPH '93. In *ACM Computer Graphics*, August 1993, 343-350.

[12] M. G. Pandy, F. E. Zajac, E Sim, and W. S. Levine. An Optimal Control Model for Maximum-Height Human Jumping. *J. Biomechanics*, 23, 12, 1185-1198, 1990.

[13] M. H. Raibert and J. K. Hodgins. Animation of dynamic legged locomotion. *Proceedings of SIGGRAPH '91*, In *ACM Computer Graphics*, 25, 4 (July 1991), 349-358.

[14] A. J. Stewart and J. F. Cremer. Beyond Keyframing: An Algorithmic Approach to Animation. *Proceedings of Graphics Interface '92*, 273-281, 1992.

[15] A. F. Vakakis, J. W. Burdick, and T. K. Caughey. An Interesting Strange Attractor in the Dynamics of a Hopping Robot. *The International Journal of Robotics Research*, 10, 6, 1991, 606-618.

[16] M. van de Panne, E. Fiume, and Z. G. Vranesic. Reusable Motion Synthesis Using State-Space Controller. *Proceedings of SIGGRAPH '90.* In *ACM Computer Graphics*, 24, 4 (August 1990), 225-234.

[17] M. van de Panne and E. Fiume. Sensor-Actuator Networks. *Proceedings of SIGGRAPH '93*, In *ACM Computer Graphics*, August 1993, 335-342.

[18] M. van de Panne, E. Fiume, and Z. Vranesic. Physically-Based Modeling and Control of Turning. *Computer Vision, Graphics, and Image Processing: Graphical Models and Image Processing,* 55, 6 (Nov. 1993), 507-521.

[19] A. Witkin and M. Kass. Spacetime Constraints. *Proceedings of SIGGRAPH '88.* In *ACM Computer Graphics*, 22, 4 (August 1988), 159-168.